CS 577: Introduction to Algorithms	Homework 9
Out: 11/24/10	**Due: 12/06/10**

Ground Rules

- **Grading.** You will be graded on the correctness as well as clarity of your solutions. You are required to prove any claims that you make. In particular, when you are asked to design an algorithm, you must argue its correctness as well as running time. You may use without proof any theorems proven in class or in the textbook, as long as you state and cite them properly.
- **Collaboration.** You may work on and submit solutions in pairs.
- **Lateness.** Homework is due promptly at the start of class. Late homework will receive zero credit.
- **Extra credit questions.** Extra credit questions will not directly contribute to your score. However, they will be taken into account in the final grading and may improve your overall grade if your total score is close to the boundary between two grades. Furthermore they are fun to solve and improve your understanding of the course.
- Start working on your homework early. Plan your work in such a way that you have the opportunity to put some problems on the back burner for a while and revisit them later. Good luck!

Problems

1. **(3+4 points.)** Randomness can be very useful when we want to verify the correctness of a program. In this question we will see a basic example of how randomness helps in generating good test cases. Suppose we are given $n \times n$ matrices A, B, and C, and wish to check whether $AB = C$. One way to do this is to just multiply A and B out and compare the result to C. Using fast matrix multiplication (Strassen's algorithm), this takes $O(n^{2.81})$ time. We will now develop an $O(n^2)$ time test that has high accuracy.

   (a) Let r be an $n \times 1$ random vector, where each entry is independently 0 with probability $1/2$ and 1 with probability $1/2$. Prove that for two $n \times n$ matrices X and Y with $X \neq Y$, the probability that $Xr = Yr$ is at most $1/2$. Note that X and Y are fixed matrices, not random matrices.

   Hint: Consider the non-zero matrix $(X - Y)$ and prove that Pr$[(X - Y)r = 0]$ is less than $1/2$.

   (b) For any given $\epsilon \in [0, 1/2]$, use the above to design an $O(n^2 \log(1/\epsilon))$ time algorithm for checking whether $AB = C$, that returns the correct answer with probability at least $1 - \epsilon$.

   Hint: Prove the $\epsilon = 1/2$ case first.

2. **(4+4 points.)** Recall the vertex cover problem from Homework 2: given a graph G with vertex set V and edge set E, the problem is to find a subset S of V of the smallest possible size such that every edge in E is incident on at least one vertex in S. Such a set S is called a vertex cover. In Homework 2 we developed an algorithm for solving this problem exactly when G is a tree. In general graphs this problem is NP-hard. We will develop a randomized algorithm that produces a nearly optimal solution.

   The algorithm is as follows:

   - Start with $S = \emptyset$
   - While S is not a vertex cover:
     - Select an arbitrary edge e not covered by S, and select an end-point of e at random (both end-points being equally likely). Add this end-point to S.
   - Output S

We will now analyze this algorithm. Let the optimal vertex cover be S^*, and let F denote the set of edges that the algorithm selects within the while loop. For $v \in S$, let $E(v)$ denote the set of edges incident on v in the graph.

(a) Prove that for any $v \in V$, the expected number of edges incident on v that the algorithm selects is at most 2. That is, $\mathrm{E}[|E(v) \cap F|] \leq 2$.

(b) Use (a) to prove that the expected size of S is at most $2|S^*|$.

Hint: $|S|$ is exactly equal to $|F|$. Every edge is incident on at least one node in S^* and (a) implies that $E[|E(v) \cap F|] \leq 2$ for all $v \in S^*$.

3. **(Extra credit.)** In this question, you are given n points each picked independently uniformly at random from the two-dimensional unit square $[0, 1]^2$. We say that a point (x_1, y_1) dominates another one (x_2, y_2), if $x_1 > x_2$ and $y_1 > y_2$. A point is undominated if no other point dominates it. Compute the expected number of undominated points.